

Teoría de Lenguajes

Primer Parcial

Segundo cuatrimestre de 2019

Apagar los celulares.

Hacer cada ejercicio en hojas separadas.

Poner nombre, número de orden y número de página en cada ejercicio.

Justificar todas las respuestas.

El examen es a libro abierto.

Se aprueba con al menos 65 puntos.

1. (25 pts)

Sea $L_1 = \{\omega \in \{0,1,2\}^* \mid \omega = d_1d_2\dots d_n, n \geq 0, (n \geq 2) \Rightarrow d_n = \max\{d_1, d_2, \dots, d_{n-1}\}\}$. Determinar si existe una expresión regular que denote L_1 . De existir, exhibir alguna. De no existir, probarlo.

2. (25 pts)

Sea $L_2 = \{a^m b^n c^r \mid m \geq n \vee n \leq r\}$. Determinar si L_2 puede ser reconocido por algún autómata finito. De existir, exhibir alguno. De no existir, probarlo.

3. (25 pts)

Sea L_3 el lenguaje sobre el alfabeto $\{ (,), [,] \}$ de las cadenas de paréntesis y corchetes balanceados tales que no tengan ningún anidamiento de más de 2 paréntesis seguidos (pero sí pueden alternar paréntesis con corchetes ilimitadamente y anidar corchetes ilimitadamente).

Ejemplos de cadenas válidas:

$()()$
 $()()([()])$
 $([[([[]])]])$
 $()[[[[]]]]$
 $([[[([()])()]]])$

Ejemplos de cadenas inválidas¹:

$((()))$	//anida 3 paréntesis
$([[([()])]])$	//anida 3 paréntesis
$([]((())))$	//anida 4 paréntesis
$([]([()([()]))])$	//anida 4 paréntesis
$([])$	//no balanceado

Dar una gramática libre de contexto para L_3 .

4. (25 pts)

Dar un autómata de pila determinístico que acepte el lenguaje L_3 .

¹Los comentarios a la derecha no forman parte de las cadenas.

Ejercicio 1

★ Primero veamos cómo caracterizar L_1 .

- $\lambda \in L_1$, ya que n puede ser 0.
- $0 \in L_1, 1 \in L_1, 2 \in L_1$, ya que $d_n = d_1 = \max\{d_i\}$.
- Si $n \geq 2$, d_n debe ser el máximo entre todos los dígitos anteriores, pero no hay ninguna restricción sobre los $d_i, i < n$. Lo que hay que asegurar es que
 - si $\max\{d_1, \dots, d_{n-1}\} = 0 \Rightarrow d_n = 0$
 - si $\max\{d_1, \dots, d_{n-1}\} = 1 \Rightarrow d_n = 1$
 - si $\max\{d_1, \dots, d_{n-1}\} = 2 \Rightarrow d_n = 2$

1	2	3	4	5
21	25	25	23	94

★ Dicho esto, podemos plantear estados que caractericen la subcadena recorrida hasta ahora según los dígitos ya leídos y el último dígito procesado en relación a ellos.

- q_0 : o bien no he leído nada, o el máximo dígito ^{de d_1, \dots, d_{n-1}} que he leído es 0 y el último también es 0; acepto.
- q_1 : el máximo dígito que

★ Ahora veamos expresiones regulares para caracterizar la forma que puede tener $d_1 \dots d_{n-1}, n \geq 2$:

- Si el máximo dígito es 0, sólo puede haber 0s, o sea que $d_1 \dots d_{n-1}$ es una cadena de uno o más 0s: 0^+
- Si el máximo dígito es 1, hay como mínimo un 1 y puede estar acompañado de 0s y de 1s en cualquier orden, tanto a su derecha como a su izquierda: $(011)^*1(011)^*$

(La expresión regular 1 representa un 1 obligatorio y $(011)^*$ representa cualquier secuencia de 0s y 1s que pueden o no estar mezclados).

- Si el máximo dígito es 2, la situación es análoga a la anterior y $d_1 \dots d_{n-1}$ es $(01112)^*2(01112)^*$

★ Teniendo ERs para las distintas formas de $d_1 \dots d_{n-1}$, podemos concatenarlas con el dígito que sí o sí tiene que ser d_n :

$0^+0, (011)^*1(011)^*1, (01112)^*2(01112)^*2$

★ Los casos en los que $n=0$ o $n=1$ son más sencillos: si $n=0$, la ER λ alcanza. Si $n=1$, la cadena es algo de $\{0\} \cup \{1\} \cup \{2\}$. En ERs, la unión se representa con $|$, por lo que la ER para $n=1$ es $0|1|2$.

★ Entonces, L_1 puede entenderse como la unión de lenguajes

$$\underbrace{L(\lambda)}_{n=0} \cup \underbrace{L(0112)}_{n=1, d_1} \cup \underbrace{L(0^+0)}_{n \geq 2, d_n=0} \cup \underbrace{L((011)^*1(011)^*1)}_{n \geq 2, d_n=1} \cup \underbrace{L((0112)^*2(0112)^*2)}_{n \geq 2, d_n=2}$$

★ Como $|$ denota la unión, podemos escribir

$$L = L(\lambda | 0112 | 0^+0 | (011)^*1(011)^*1 | (0112)^*2(0112)^*2)$$

★ Simplificamos la ER reordenando los términos, ya que $|$ es conmutativo (porque también lo es la unión de conjuntos / lenguajes):

$$\lambda | 010^+011 | (011)^*1(011)^*1 | 2 | (0112)^*2(0112)^*2 =$$

$$\lambda | (0^+0) | (\lambda | (011)^*1(011)^*1 | 1 | (\lambda | (0112)^*2(0112)^*2))$$

$$\lambda | (\lambda | 0^+) | 011 | (011)^*1(011)^*1 | 2 | (0112)^*2(0112)^*2 =$$

$$\lambda | \underbrace{0^+0}_{0^+} | 11 | (011)^*1(011)^*1 | 2 | (0112)^*2(0112)^*2 =$$

$$\lambda | 0^+ = 0^*$$

$$\underline{0^*111(011)^*1(011)^*1 | 2 | (0112)^*2(0112)^*2}$$

★ Para esta simplificación, usé las siguientes propiedades de ERs vistas en clase:

$$0^*0 = 00^* = 0^+$$

$$0^* = \lambda | 0^+ = 0^+ | \lambda$$

$$e^*e = ee^* = e^+$$

$$e^* = \lambda | e^+ = e^+ | \lambda$$

$$f | e | g = (f | g) | e$$

(siendo e, f y g expresiones regulares).

se justificaría un poco más si se partiera de un AX y después se convirtiera.

Ejercicio 2

$$L_2 = \{a^m b^n c^r \mid m \geq n \vee n \leq r\}$$

(*) Ver carilla siguiente

★ Por lo visto en clase, existe una expresión regular para denotar L_2 si y sólo si L_2 es regular. Vamos a demostrar por absurdo que no lo es, y que por lo tanto no puede existir una ER que lo denote.

★ Supongamos que L_2 es regular. Entonces, por una propiedad vista en clase, también lo es su complemento L_2^c . L_2^c es difícil de caracterizar, puesto que incluye cadenas de formas muy variadas, pero en particular podemos afirmar lo siguiente sobre w de la forma $a^m b^n c^r$:

$$m \geq n \vee n \leq r \Rightarrow w \in L_2 \Leftrightarrow w \notin L_2^c$$

$$m < n \wedge n > r \Rightarrow w \in L_2^c$$

Esto va a usarse más adelante.

★ Por lo visto en clase, si L_2^c es regular entonces cumple el lema de pumping:
 $\exists p \in \mathbb{N} / \forall w \in L_2^c, |w| \geq p: \exists x, y, z / w = xyz \wedge |xy| \leq p \wedge |y| > 0 \wedge \forall i \in \mathbb{N}_0: xy^i z \in L_2^c$.

★ Sea p la constante de pumping y sea $w = a^p b^{p+1} c^p$.

$$|w| = p + p + 1 + p = 3p + 1 \geq p$$

$$p < p + 1 \Rightarrow w \in L_2^c$$

$\therefore w$ cumple las hipótesis del lema de pumping, por lo que deben existir $x, y, z / w = xyz \wedge |xy| \leq p \wedge |y| > 0 \wedge \forall i \in \mathbb{N}_0: xy^i z \in L_2^c$.

★ Sean $x, y, z / w = xyz, |xy| \leq p, |y| > 0$. Veamos qué forma tienen x, y y z .

• Los primeros p símbolos de w son a 's y $|xy| \leq p \Rightarrow xy = a^k, k \leq p$.

$$xy = a^k \Rightarrow y = a^j, 0 < j \leq k$$

$$xyz = \underbrace{a^{k-j}}_x \underbrace{a^j}_y \underbrace{a^{p-k} b^{p+1} c^p}_z$$

★ Sabiendo esto, podemos escribir $xy^i z, i \in \mathbb{N}_0$ como:

$$\begin{aligned} xy^i z &= a^{k-j} (a^j)^i a^{p-k} b^{p+1} c^p = a^{k-j+ij} a^{p-k} b^{p+1} c^p \\ &= a^{p-k+k+(i-1)j} b^{p+1} c^p = a^{p+(i-1)j} b^{p+1} c^p \end{aligned}$$

★ Sea $i=2$.

Como L_2^c es regular y cumple el lema de pumping, $xy^2 z \in L_2^c$.

$$\star xy^2z = a^{p+(2-1)j} b^{p+1} c^p = \underline{a^{p+j} b^{p+1} c^p}. \quad \checkmark$$

$\star j \geq 1 \Rightarrow p+j \geq p+1 \Rightarrow xy^2z$ es de la forma $a^m b^n c^r$, $m \geq n \vee n \leq r \Rightarrow \underline{xy^2z \in L_2}$. \checkmark

$\Rightarrow \underline{xy^2z \notin L_2^c}$. \checkmark Absurdo, ya que habíamos afirmado que $xy^2z \in L_2^c$.

El absurdo vino de suponer que L_2 era regular, de lo cual concluimos que L_2 no es regular.

\therefore No existe un autómata finito que reconozca L_2 . \checkmark

(*) En vez de decir que existe una ER si y sólo si L_2 es regular, debería decir que existe un autómata finito que acepta L_2 si y sólo si L_2 es regular (las afirmaciones son equivalentes, pero esta última es la que pide la consigna). \checkmark

Ejercicio 3

* Para asegurar que se cumpla la restricción sobre el anidamiento de paréntesis, la gramática necesita por lo menos tres símbolos terminales:

- uno del que siempre se deriven producciones sin paréntesis ^{afuera de todo} ↑
- uno del que siempre se deriven producciones en las que el máximo anidamiento de paréntesis afuera de todo sea un nivel
- uno del que siempre se deriven producciones en las que el máximo anidamiento de paréntesis afuera de todo sean dos niveles

Los símbolos podrían producir cadenas en las que no se alcance este máximo, pero el invariante va a ser usado para saber que se pueden agregar paréntesis sin llegar nunca a tres anidamientos seguidos.

* Las siguientes reglas de producción cumplen lo que acabamos de ver:

$S \rightarrow RITI U$ (S va a ser nuestro símbolo inicial)

$R \rightarrow [R] | [T] | [U] | \lambda | RR$

$T \rightarrow (R) | RT | TR | TT$

$U \rightarrow (T) | U S | S U$

- R siempre produce cosas que o bien están rodeadas por corchetes, o son concatenaciones de cadenas rodeadas por corchetes o son λ .
- T siempre produce cosas que tienen, como máximo, un ^{par} anidamiento de paréntesis (* en los extremos de la cadena (o pueden no tener ninguno y los paréntesis pueden estar en la parte interna de la cadena). Esto se debe a que siempre que se agregan paréntesis derivando T , éstos rodean R , que no tiene paréntesis externos.
- U es análogo a T pero con anidamientos de dos paréntesis seguidos. Como ninguna regla de la gramática permite rodear de paréntesis un U , se cumple la restricción de los anidamientos de paréntesis.
- Observemos que R sólo se concatena con R , mientras que T se concatena con R o con T . Esto es porque sólo queremos que puedan concatenarse con cosas que tengan, como máximo, la misma cantidad de paréntesis externos que pueden tener ellas, para que se mantenga el invariante si estos símbolos están adentro de otro par de paréntesis (como T en U y R en T).
- U puede concatenarse con cualquier cosa (pues S deriva en cualquier cosa) porque nunca va a aparecer entre paréntesis.

★ Así, la gramática resultante es la siguiente:

$$G = \langle V_T, V_N, P, S \rangle$$

$$V_T = \{ (,), [,] \}$$

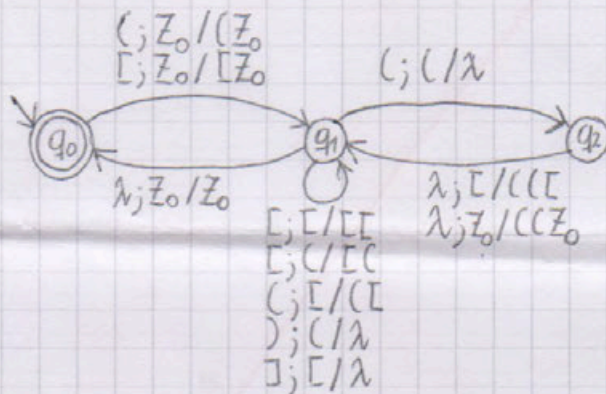
$$V_N = \{ S, R, T, U \}$$

$$P = \{ \begin{array}{l} S \rightarrow R T U, \\ R \rightarrow [R] | [T] | [U] | \Lambda I R R, \\ T \rightarrow (R) | T T | R T | T R, \\ U \rightarrow (T) | S U | U S \end{array} \}$$

Ejercicio 4

- ★ El APD debería seguir las reglas básicas para aceptar cadenas balanceadas: no puede recibir un corchete de cierre si el último símbolo de apertura que recibió fue un paréntesis (o viceversa) y al final tiene que estar Z_0 en el tope de la pila (el autó-mata que acepta cadenas de paréntesis y corchetes balanceados fue visto en la práctica así que no voy a ahondar).
- ★ Además, cuando recibe un paréntesis de apertura, tiene que chequear que éste no abra un tercer par anidado consecutivo. Así en el tope hay un corchete, está fuera de peligro. Si hay un paréntesis, debe desapilarlo, trabarse si hay otro paréntesis abajo y volver a apilarlo (junto con el nuevo paréntesis) si hay otra cosa abajo.
- ★ Para los símbolos de cierre, alcanza con verificar que en el tope esté el símbolo de apertura correcto y desapilarlo.

★ APD:



$$M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, \{q_0\} \rangle$$

$$Q = \{q_0, q_1, q_2\}$$

$$\Sigma = \{[,], (,)\}$$

$$\Gamma = \{[, (, Z_0\}$$

- ★ Por más que tenga transiciones λ , el autó-mata es determinístico, ya que toma como entrada para δ el tope de la pila y para cada tope posible existe a lo sumo una transición desde cada estado. (y ninguna

Transición λ para
en tope)